IN THE CLAIMS

Please amend claims 1-16 and 21-27 as indicated below.

Please cancel without prejudice claims 17-20 and 28-32.

Please add new claims 33-39 as indicated below.

1. (Currently Amended) A method for handling ~~a number of~~ exceptions ~~within a processor~~ in a multi-~~processing~~ processor system, the method comprising:

receiving an exception within ~~the~~ a processor which is one of a plurality of processors of the multi-processor system, wherein each of the processors in the multi-processor system shares a ~~same~~ memory within the multi-processor system, wherein the memory includes a common interrupt handling vector address space shared by the plurality of the processors and a dedicated interrupt handling vector address space for each of the plurality of the processors; and

executing ~~a number of~~ one or more instructions at an address associated with a type of the received exception within ~~a~~ the common interrupt handling vector address space of the ~~same~~ memory, wherein the ~~number of~~ one or more instructions cause the processor to modify based on ~~determine~~ an identification of the processor ~~based on a query that is internal to the processor; and~~

~~modifying~~ an execution flow of the exception to execute an interrupt handler located within ~~one of a number of different~~ a respective dedicated interrupt handling vector address spaces associated with the processor.

2. (Currently Amended) The method of claim 1, wherein each processor in the multi-

processor system executes one of a ~~number~~ plurality of operating systems, and wherein at least two of the operating systems are different.

3. (Currently Amended) The method of claim 2, wherein each of the ~~number of~~ operating systems is associated with ~~one of the number of differenta~~ dedicated interrupt handling vector address spaces associated with a processor executing the respective operating system.

4. (Currently Amended) The method of claim 1, further comprising determining the identification of the processor ~~wherein the query that is internal to the processor includes~~ by reading a bit ~~within~~ of a register ~~that internal to~~within the processor without having to access the memory.

5. (Currently Amended) The method of claim 4, wherein the register ~~is not dedicated to determining the identification of the processor~~is part of a cache throttling register of the processor, and wherein a dedicated bit of the register is used to indicate the identification of the processor while at least a portion of remaining bits is used for cache throttling purposes.

6. (Currently Amended) The method of claim ~~1~~4, ~~further comprising~~wherein determining the identification of the processor comprises:

communicating ~~during~~ an initialization of the processor~~, based communications~~ with a memory controller ~~that is coupled between~~coupling the processors ~~in the multiple processor system~~ and the ~~same~~ memory to retrieve the identification of the processor from the memory controller; and

storing the retrieved identification of the processor in the register within the processor.

7. (Currently Amended) The method of claim 1, wherein each ~~of the number of exceptions~~

~~received by the different processors execute instructions at an address within the common interrupt handling vector address space of the same memory~~entry of the common interrupt handling vector space is associated with a different type of exceptions of the multi-processor system, and wherein a result of executing the instructions in the common interrupt handling vector space and the identification of the processor receiving the exception determine the dedicated interrupt handling vector space associated with the processor.

8. (Currently Amended) ~~A method comprising:~~

~~receiving an exception within a processor, wherein the processor is included in a multi-processor system, wherein each processor in the multi-processor system executes one of a number of operating systems, wherein each processor in the multi-processor system shares a same memory and wherein the same memory includes a common interrupt handling address space and a number of different interrupt handling address spaces associated with each of the different processors in the multi-processor system;~~

~~determining the type of exception received within the processor;~~

~~executing a number of instructions at an address within the common interrupt handling address space of the same memory, wherein the number of instructions cause the processor to read a bit within an internal register to determine an identification of the processor in the multi-processor system; and~~

~~modifying execution flow of the exception to execute an interrupt handler located within one of the number of different interrupt handling address spaces.~~ The method of claim 7, further comprising:

determining a type of the exception received by the processor;

determining an entry of the common interrupt handling vector space based on the determined type of the exception; and

executing the one or more instructions stored at the determined entry to determine the dedicated interrupt handling vector space associated with the processor based on the identification of the processor.

9. (Currently Amended) The method of claim 8, further comprising executing an interrupt service routine based on the determined entry of the dedicated interrupt handling vector space of the processor to handle the exception ~~wherein the internal register is not dedicated to determining the identification of the processor~~.

10. (Currently Amended) The method of claim ~~8~~6, wherein ~~each of the number of operating systems is associated with one of the number of different interrupt handling address spaces~~the memory controller includes a plurality of interfaces and each of the interfaces is coupled to one of the plurality of processor, and wherein when each of the processors is being initialized during an initialization period of the multi-processor system, each of the processors transmits a signal via a respective interface to the memory controller to identify the respective processor.

11. (Currently Amended) The method of claim ~~8~~10, wherein in response to the signal received from each processor, the memory controller stores an identification of a respective processor within the memory controller, such that the identification of the respective processor is queried from the memory controller during runtime without having to access the memory ~~further comprising determining the identification of the processor during initialization of the processor, based communications with a memory controller that is coupled between the processors in the multiple processor system and the same memory~~.

12. (Original) A system comprising:

a plurality of processors including a first and a second processors;

a memory that includes:

a common exception handling vector address space~~;~~ shared by the plurality of processors, and

a ~~number~~ plurality of exception handling vector address spaces each associated with each of the plurality of processors, including a first and a second exception handling vector address spaces associated with the first and second processors respectively;

a memory controller coupled to the memory and the plurality of processors, wherein the~~;~~ ~~a~~ first processor ~~coupled to the memory controller, wherein the first processor~~ is to execute a first operating system~~,~~; and

wherein the~~a~~ second processor ~~coupled to the memory controller, wherein the second processor~~ is to execute a second operating system, the second processor to execute ~~a number of~~ one or more instructions in the common exception handling vector address space upon receipt of an exception, wherein the ~~number of~~one or more instructions cause the second processor to modify based on an identification of the second processor an execution flow of the exception to execute an interrupt handler located within a respective dedicated interrupt handling vector address spaces associated with the second processor ~~determine an identification of the second processor based on a query that is internal to the second processor, wherein the first processor and the second processor share the memory~~.

13. (Currently Amended) The system of claim 12, wherein the first and second operating

system are different ~~the first operating system and the second operating system are each associated with one of the number of different interrupt handling vector address spaces~~.

14. (Currently Amended) The system of claim 12, wherein the second processor further comprises an internal register ~~and wherein the query that is internal to the second processor includes~~ to determine the identification of the second processor by reading a bit within the internal register.

15. (Currently Amended) The system of claim 14, wherein the internal register is ~~not dedicated to determining the identification of the second processor~~part of a cache throttling register of the second processor, and wherein a dedicated bit of the internal register is used to indicate the identification of the second processor while at least a portion of remaining bits are used for cache throttling purposes.

16. (Currently Amended) The system of claim ~~12~~14, wherein the identification of the second processor is determined by ~~the second processor is to determine the identification of the second processor during initialization, based communications with the memory controller~~

communicating during an initialization of the second processor with the memory controller to retrieve the identification of the second processor from the memory controller, and

storing the retrieved identification of the second processor in the internal register within the second processor.

17. – 20. (Canceled)

21. (Currently Amended) A machine-readable medium that provides instructions for handling

~~a number of~~ exceptions within ~~a processor in~~ a multi-processor~~ing~~ system, which when executed by a machine, causes the machine to perform operations comprising:

receiving an exception within a processor which is one of a plurality of processors of the multi-processor system, wherein each of the processors in the multi-processor system shares a memory within the multi-processor system, wherein the memory includes a common interrupt handling vector address space shared by the plurality of the processors and a dedicated interrupt handling vector address space for each of the plurality of the processors; and

executing one or more instructions at an address associated with a type of the received exception within the common interrupt handling vector address space of the memory, wherein the one or more instructions cause the processor to modify based on an identification of the processor an execution flow of the exception to execute an interrupt handler located within a respective dedicated interrupt handling vector address spaces associated with the processor.

~~receiving an exception within the processor, wherein each processor in the multi-processor system shares a same memory;~~

~~executing a number of instructions at an address within a common interrupt handling vector address space of the same memory, wherein the number of instructions cause the processor to determine an identification of the processor based on a query that is internal to the processor; and~~

~~modifying execution flow of the exception to execute an interrupt handler located within one of a number of different interrupt handling vector address spaces.~~

22. (Currently Amended) The machine-readable medium of claim 21, wherein each processor in the multi-processor system executes one of a ~~number~~ plurality of operating systems, and

wherein at least two of the operating systems are different.

23. (Currently Amended) The machine-readable medium of claim 22, wherein each of the ~~number of~~ operating systems is associated with ~~one of the number of different~~a dedicated interrupt handling vector address spaces associated with a processor executing the respective operating system.

24. (Currently Amended) The machine-readable medium of claim 21, further comprising determining the identification of the processor by ~~wherein the query that is internal to the processor includes~~ reading a bit ~~within~~ of a register ~~that internal to~~within the processor without having to access the memory.

25. (Currently Amended) The machine-readable medium of claim 24, wherein the register is part of a cache throttling register of the processor, and wherein a dedicated bit of the register is used to indicate the identification of the processor while at least a portion of remaining bits of the register is used for cache throttling purposes ~~is not dedicated to determining the identification of the processor~~.

26. (Currently Amended) The machine-readable medium of claim 21, ~~further comprising~~wherein determining the identification of the processor comprises:

communicating during an initialization of the processor~~, based communications~~ with a memory controller ~~that is coupled between~~coupling the processors ~~in the multiple processor system~~ and the ~~same~~ memory to retrieve the identification of the processor from the memory controller; and

storing the retrieved identification of the processor in the register within the processor.

27. (Currently Amended) The machine-readable medium of claim 21, wherein each entry of the common interrupt handling vector space is associated with a different type of exceptions of the multi-processor system, and wherein a result of executing the instructions in the common interrupt handling vector space and the identification of the processor receiving the exception determine the dedicated interrupt handling vector space associated with the processor ~~of the number of exceptions received by the different processors execute instructions at an address within the common interrupt handling vector address space of the same memory~~.

28. – 32. (Canceled)

33. (New) The method of claim 2, wherein the multi-processor system is a part of a network element interfacing a first network and a second network different than the first network.

34. (New) The method of claim 33, wherein the multi-processor system is implemented as a control card of the network element, wherein the network element further comprises a first line card coupled the first network and a second line card coupled to the second network, and wherein the control card is to control routing network traffic between the first and second networks via the first and second line cards respectively.

35. (New) The method of claim 34, wherein the first network is a network provider network and the second network is a service provider network.

36. (New) The method of claim 34, wherein an operating system executed by the first processor is a real-time operating system to handle routing network traffic and an operating system executed by the second processor is a non-real-time operating system to handle at least one of provisioning and configuration of the network element.

37. (New) A network element for routing traffic between networks, comprising:

a first line card coupled to a first network;

a second line card coupled to a second network different than the first network;

a control card coupled to the first and second line cards for routing network traffic between the first and second networks via the first and second line cards, the control card including

a plurality of processors including a first and a second processors,

a memory including

a common exception handling vector address space shared by the plurality of processors, and

a plurality of exception handling vector address spaces each associated with each of the plurality of processors, including a first and a second exception handling vector address spaces associated with the first and second processors respectively,

a memory controller coupled to the memory and the plurality of processors,

wherein the first processor is to execute a first operating system, and

wherein the second processor is to execute a second operating system and to execute one or more instructions in the common exception handling vector address space upon receipt of an exception, wherein the one or more instructions cause the

second processor to modify based on an identification of the second processor an execution flow of the exception to execute an interrupt handler located within a respective dedicated interrupt handling vector address spaces associated with the second processor.

38. (New) The network element of claim 37, wherein the first network is a network provider network and the second network is a service provider network.

39. (New) The network element of claim 37, wherein the first operating system is a real-time operating system to handle routing network traffic and the second operating system is a non-real-time operating system to handle at least one of provisioning and configuration of the network element.